School of Data and Computer Science, Sun Yat-sen University

E08 Bayesian Network

16337102 Zilin Huang

October 26, 2018

Contents

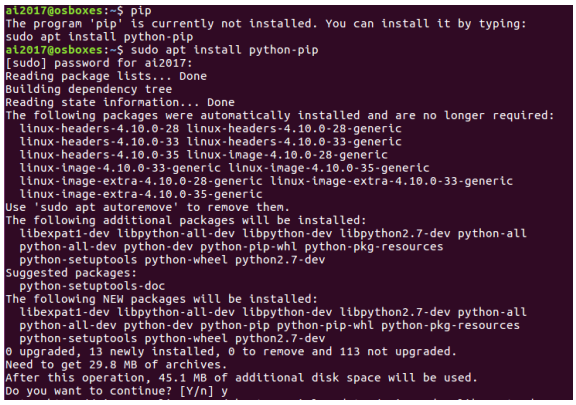
1 Pomegranate Installation

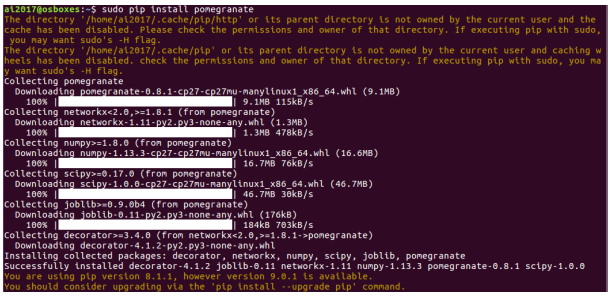
Under Linux:

1. Install `python` first (**python 2**, not python 3).

2. Run `sudo apt-get install python-pip` to install `pip`.

3. Run `sudo pip install pomegranate` to install `pomegranate`.

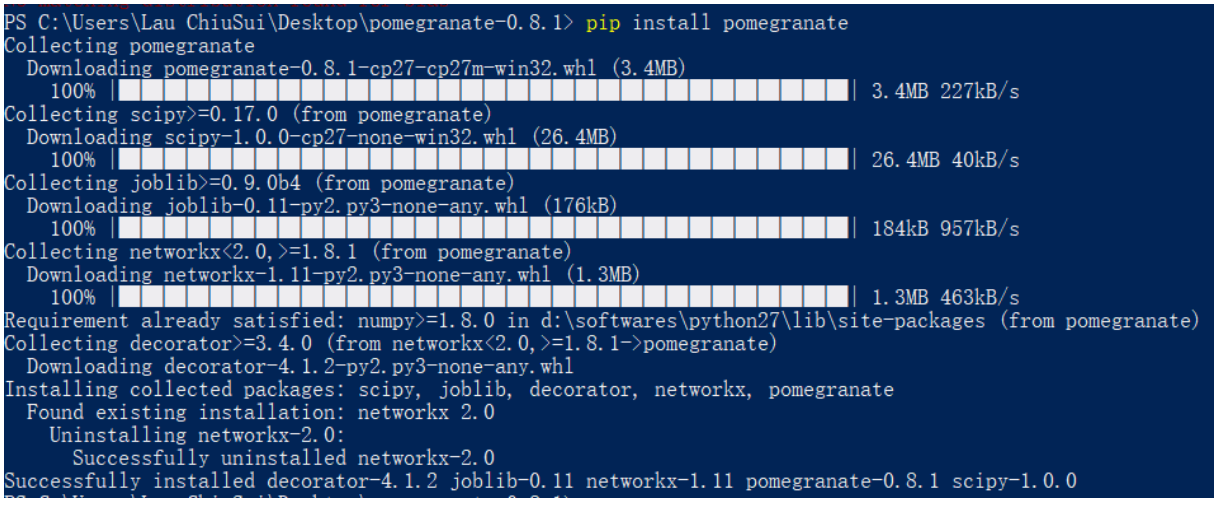




Under Windows

You can also run `pip install pomegranate` if you have installed `pip`. If you don't know how to install `pip`, please click `https://jingyan.baidu.com/article/e73e26c0d94e0524adb6a7ff.html`.

For more, please click the homepage of Pomegranate - `https://github.com/jmschrei/pomegranate` for help.



2 Building Bayesian Network

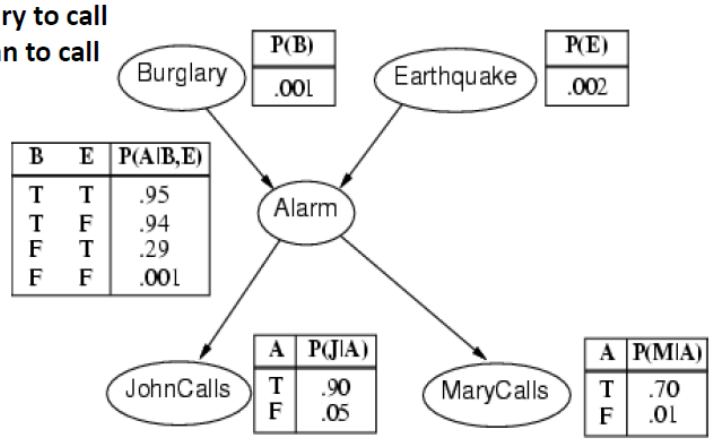
Please refer to `Tutorial_4_Bayesian_Networks.pdf`. I will explain it in class.

3 Tasks

3.1 Burglary

- **A burglary can set the alarm off**
- **An earthquake can set the alarm off**
- **The alarm can cause Mary to call**
- **The alarm can cause John to call**

Note that these tables only provide the probability that Xi is true.
(E.g., Pr(A is true|B,E))
The probability that Xi is false is 1- these values



Please code to calculate:

1. $P(A)$
2. $P(J\overline{M})$
3. $P(A|J\overline{M})$
4. $P(B|A)$
5. $P(B|J\overline{M})$
6. $P(J\overline{M}|\overline{B})$

```
P(Alarm) =
0.002516442

P(J&&~M) =
0.050054875461

P(A |J&&~M) =
0.0135738893313

P(B |A) =
0.373551228282

P(B |J&&~M) =
0.0051298581334

P(J&&~M |~B) =
0.049847949
```

3.2 Diagnosing

Variables and their domais

(1) PatientAge:['0-30','31-65','65+']

(2) CTScanResult:['Ischemic Stroke','Hemmorraghic Stroke']

(3) MRIScanResult: ['Ischemic Stroke','Hemmorraghic Stroke']

(4) StrokeType: ['Ischemic Stroke','Hemmorraghic Stroke', 'Stroke Mimic']

(5) Anticoagulants: ['Used','Not used']

(6) Mortality:['True', 'False']

(7) Disability: ['Negligible', 'Moderate', 'Severe']

CPTs

Note: [CTScanResult, MRIScanResult,StrokeType] means:

P(StrokeType='...' | CTScanResult='...' $\wedge$ MRIScanResult='...')

(1)

[PatientAge]

['0-30', 0.10],

['31-65', 0.30],

['65+', 0.60]

(2)

[CTScanResult]

['Ischemic Stroke',0.7],

['Hemmorraghic Stroke',0.3]

(3)

[MRIScanResult]

['Ischemic Stroke',0.7],

['Hemmorraghic Stroke',0.3]

(4)

```
[Anticoagulants]

['Used',0.5],
['Not used',0.5]

(5)
[CTScanResult, MRIScanResult,StrokeType])

['Ischemic Stroke','Ischemic Stroke','Ischemic Stroke',0.8],
['Ischemic Stroke','Hemmorraghic Stroke','Ischemic Stroke',0.5],
['Hemmorraghic Stroke','Ischemic Stroke','Ischemic Stroke',0.5],
['Hemmorraghic Stroke','Hemmorraghic Stroke','Ischemic Stroke',0],

['Ischemic Stroke','Ischemic Stroke','Hemmorraghic Stroke',0],
['Ischemic Stroke','Hemmorraghic Stroke','Hemmorraghic Stroke',0.4],
['Hemmorraghic Stroke','Ischemic Stroke','Hemmorraghic Stroke',0.4],
['Hemmorraghic Stroke','Hemmorraghic Stroke','Hemmorraghic Stroke',0.9],

['Ischemic Stroke','Ischemic Stroke','Stroke Mimic',0.2],
['Ischemic Stroke','Hemmorraghic Stroke','Stroke Mimic',0.1],
['Hemmorraghic Stroke','Ischemic Stroke','Stroke Mimic',0.1],
['Hemmorraghic Stroke','Hemmorraghic Stroke','Stroke Mimic',0.1],

(6)
[StrokeType, Anticoagulants, Mortality]

['Ischemic Stroke', 'Used', 'False',0.28],
['Hemmorraghic Stroke', 'Used', 'False',0.99],
['Stroke Mimic', 'Used', 'False',0.1],
['Ischemic Stroke','Not used', 'False',0.56],
['Hemmorraghic Stroke', 'Not used', 'False',0.58],
['Stroke Mimic', 'Not used', 'False',0.05],
```

```
['Ischemic Stroke',  'Used' ,'True',0.72],
['Hemmorraghic Stroke', 'Used', 'True',0.01],
['Stroke Mimic', 'Used', 'True',0.9],
['Ischemic Stroke',  'Not used' ,'True',0.44],
['Hemmorraghic Stroke', 'Not used', 'True',0.42 ],
['Stroke Mimic', 'Not used', 'True',0.95]
```

(7)

```
[StrokeType, PatientAge, Disability]

['Ischemic Stroke',   '0-30','Negligible', 0.80],
['Hemmorraghic Stroke', '0-30','Negligible', 0.70],
['Stroke Mimic',        '0-30', 'Negligible',0.9],
['Ischemic Stroke',     '31-65','Negligible', 0.60],
['Hemmorraghic Stroke', '31-65','Negligible', 0.50],
['Stroke Mimic',        '31-65', 'Negligible',0.4],
['Ischemic Stroke',     '65+'  , 'Negligible',0.30],
['Hemmorraghic Stroke', '65+'  , 'Negligible',0.20],
['Stroke Mimic',        '65+'  , 'Negligible',0.1],

['Ischemic Stroke',     '0-30' ,'Moderate',0.1],
['Hemmorraghic Stroke', '0-30' ,'Moderate',0.2],
['Stroke Mimic',        '0-30' ,'Moderate',0.05],
['Ischemic Stroke',     '31-65','Moderate',0.3],
['Hemmorraghic Stroke', '31-65','Moderate',0.4],
['Stroke Mimic',        '31-65','Moderate',0.3],
['Ischemic Stroke',     '65+'  ,'Moderate',0.4],
['Hemmorraghic Stroke', '65+'  ,'Moderate',0.2],
['Stroke Mimic',        '65+'  ,'Moderate',0.1],

['Ischemic Stroke',     '0-30' ,'Severe',0.1],
['Hemmorraghic Stroke', '0-30' ,'Severe',0.1],
['Stroke Mimic',        '0-30' ,'Severe',0.05],
```

```
['Ischemic Stroke',      '31-65','Severe',0.1],
['Hemmorraghic Stroke', '31-65','Severe',0.1],
['Stroke Mimic',         '31-65','Severe',0.3],
['Ischemic Stroke',      '65+'  ,'Severe',0.3],
['Hemmorraghic Stroke', '65+'  ,'Severe',0.6],
['Stroke Mimic',         '65+'  ,'Severe',0.8]
```

Calculation

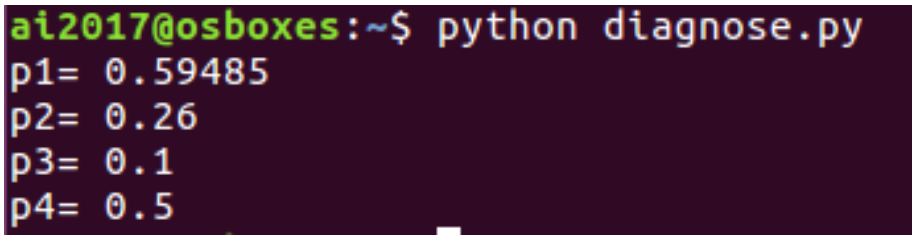
Please code to calculate the following probability value:

p1 = P(Mortality='True' | PatientAge='31-65' ∧ CTScanResult='Ischemic Stroke')

p2 = P(Disability='Moderate' | PatientAge='65+' ∧ MRIScanResult='Hemmorraghic Stroke')

p3 = P(StrokeType='Stroke Mimic' | PatientAge='65+' ∧ CTScanResult='Hemmorraghic Stroke' ∧ MRIScanResult='Ischemic Stroke')

p4 = P(Anticoagulants='Not used' | PatientAge='0-30')



Please solve the 2 tasks and hand in a file named E08_YourNumber.pdf, and send it to ai_2018@foxmail.com

4 Codes

Burglary.py

```
from pomegranate import *
B = DiscreteDistribution({'True': 0.001, 'False': 0.999})
E = DiscreteDistribution({'True': 0.002, 'False': 0.998})
A = ConditionalProbabilityTable(
    [['True', 'True', 'True', 0.95],
     ['True', 'True', 'False', 0.05],
     ['True', 'False', 'True', 0.94],
     ['True', 'False', 'False', 0.06],
     ['False', 'True', 'True', 0.29],
```

```
      ['False', 'True', 'False', 0.71],
      ['False', 'False', 'True', 0.001],
      ['False', 'False', 'False', 0.999]], [B, E])

J = ConditionalProbabilityTable(
    [['True', 'True', 0.90],
     ['True', 'False', 0.10],
     ['False', 'True', 0.05],
     ['False', 'False', 0.95]], [A] )

M = ConditionalProbabilityTable(
    [['True', 'True', 0.70],
     ['True', 'False', 0.30],
     ['False', 'True', 0.01],
     ['False', 'False', 0.99]], [A])

s1 = State(B, name = "B")
s2 = State(E, name = 'E')
s3 = State(A, name = 'A')
s4 = State(J, name = "J")
s5 = State(M, name = "M")

model = BayesianNetwork("Burglary")
model.add_states(s1, s2, s3, s4, s5)
model.add_transition(s1, s3)
model.add_transition(s2, s3)
model.add_transition(s3, s4)
model.add_transition(s3, s5)

model.bake()
#P(A)
p_a = model.predict_proba({})[2].parameters[0]['True']
print('P(Alarm) = ')
```

```
print(p_a)
print()

p_jM = model.probability(['True', 'True', 'True', 'True', 'False']) + \
    model.probability(['True', 'True', 'False', 'True', 'False']) + \
    model.probability(['True', 'False', 'True', 'True', 'False']) + \
    model.probability(['True', 'False', 'False', 'True', 'False']) + \
    model.probability(['False', 'True', 'True', 'True', 'False']) + \
    model.probability(['False', 'True', 'False', 'True', 'False']) + \
    model.probability(['False', 'False', 'True', 'True', 'False']) + \
    model.probability(['False', 'False', 'False', 'True', 'False'])
print('P(J&&~M) = ')
print(p_jM)
print()

p_ajM = model.probability(['True', 'True', 'True', 'True', 'False']) + \
    model.probability(['True', 'False', 'True', 'True', 'False']) + \
    model.probability(['False', 'True', 'True', 'True', 'False']) + \
    model.probability(['False', 'False', 'True', 'True', 'False'])
p_a_jM =  p_ajM / p_jM
print('P(A|J&&~M) = ')
print(p_a_jM)
print()

p_b_a = model.predict_proba({'A': 'True'})[0].parameters[0]['True']
print('P(B|A) = ')
print(p_b_a)
print()

p_bjM = model.probability(['True', 'True', 'True', 'True', 'False']) + \
    model.probability(['True', 'True', 'False', 'True', 'False']) + \
    model.probability(['True', 'False', 'True', 'True', 'False']) + \
    model.probability(['True', 'False', 'False', 'True', 'False'])
```

```
p_b_jM = p_bjM / p_jM
print('P(B|J&&~M) = ')
print(p_b_jM)
print()

p_B_jM = 1 - p_b_jM
p_BjM = p_B_jM * p_jM
p_B = 1- model.predict_proba({})[0].parameters[0]['True']
p_jM_B = p_BjM / p_B
print('P(J&&~M|~B) = ')
print(p_jM_B)
```

Diagnosing.py

```
from pomegranate import *
PatientAge = DiscreteDistribution(
    {'0-30': 0.10, '31-65': 0.30, '65+': 0.60})
CTScanResult = DiscreteDistribution(
    {'Ischemic Stroke': 0.7, 'Hemmorraghic Stroke': 0.3})
MRIScanResult = DiscreteDistribution(
    {'Ischemic Stroke': 0.7, 'Hemmorraghic Stroke': 0.3})
Anticoagulants = DiscreteDistribution(
    {'Used': 0.5, 'Not used': 0.5})
StrokeType = ConditionalProbabilityTable(
    [['Ischemic Stroke', 'Ischemic Stroke', 'Ischemic Stroke', 0.8],
     ['Ischemic Stroke', 'Hemmorraghic Stroke', 'Ischemic Stroke', 0.5],
     ['Hemmorraghic Stroke', 'Ischemic Stroke', 'Ischemic Stroke', 0.5],
     ['Hemmorraghic Stroke', 'Hemmorraghic Stroke', 'Ischemic Stroke', 0],

     ['Ischemic Stroke', 'Ischemic Stroke', \
     'Hemmorraghic Stroke', 0],
     ['Ischemic Stroke', 'Hemmorraghic Stroke', \
     'Hemmorraghic Stroke', 0.4],
     ['Hemmorraghic Stroke', 'Ischemic Stroke', \
```

```
      'Hemmorraghic Stroke', 0.4],
      ['Hemmorraghic Stroke', 'Hemmorraghic Stroke', \
      'Hemmorraghic Stroke', 0.9],

      ['Ischemic Stroke', 'Ischemic Stroke', 'Stroke Mimic', 0.2],
      ['Ischemic Stroke', 'Hemmorraghic Stroke', 'Stroke Mimic', 0.1],
      ['Hemmorraghic Stroke', 'Ischemic Stroke', 'Stroke Mimic', 0.1],
      ['Hemmorraghic Stroke', 'Hemmorraghic Stroke', 'Stroke Mimic', 0.1]],
      [CTScanResult, MRIScanResult])
Mortality = ConditionalProbabilityTable(
    [['Ischemic Stroke', 'Used', 'False', 0.28],
     ['Hemmorraghic Stroke', 'Used', 'False', 0.99],
     ['Stroke Mimic', 'Used', 'False', 0.1],
     ['Ischemic Stroke', 'Not used', 'False', 0.56],
     ['Hemmorraghic Stroke', 'Not used', 'False', 0.58],
     ['Stroke Mimic', 'Not used', 'False', 0.05],
     ['Ischemic Stroke', 'Used', 'True', 0.72],
     ['Hemmorraghic Stroke', 'Used', 'True', 0.01],
     ['Stroke Mimic', 'Used', 'True', 0.9],
     ['Ischemic Stroke', 'Not used', 'True', 0.44],
     ['Hemmorraghic Stroke', 'Not used', 'True', 0.42],
     ['Stroke Mimic', 'Not used', 'True', 0.95]],
     [StrokeType, Anticoagulants])
Disability = ConditionalProbabilityTable(
    [['Ischemic Stroke', '0-30', 'Negligible', 0.80],
     ['Hemmorraghic Stroke', '0-30', 'Negligible', 0.70],
     ['Stroke Mimic', '0-30', 'Negligible', 0.9],

     ['Ischemic Stroke', '31-65', 'Negligible', 0.60],
     ['Hemmorraghic Stroke', '31-65', 'Negligible', 0.50],
     ['Stroke Mimic', '31-65', 'Negligible', 0.4],

     ['Ischemic Stroke', '65+', 'Negligible', 0.30],
```

```
    ['Hemmorraghic Stroke', '65+', 'Negligible', 0.20],
    ['Stroke Mimic', '65+', 'Negligible', 0.1],

    ['Ischemic Stroke', '0-30', 'Moderate', 0.1],
    ['Hemmorraghic Stroke', '0-30', 'Moderate', 0.20],
    ['Stroke Mimic', '0-30', 'Moderate', 0.05],

    ['Ischemic Stroke', '31-65', 'Moderate', 0.3],
    ['Hemmorraghic Stroke', '31-65', 'Moderate', 0.4],
    ['Stroke Mimic', '31-65', 'Moderate', 0.3],

    ['Ischemic Stroke', '65+', 'Moderate', 0.4],
    ['Hemmorraghic Stroke', '65+', 'Moderate', 0.20],
    ['Stroke Mimic', '65+', 'Moderate', 0.1],

    ['Ischemic Stroke', '0-30', 'Severe', 0.1],
    ['Hemmorraghic Stroke', '0-30', 'Severe', 0.1],
    ['Stroke Mimic', '0-30', 'Severe', 0.05],

    ['Ischemic Stroke', '31-65', 'Severe', 0.1],
    ['Hemmorraghic Stroke', '31-65', 'Severe', 0.1],
    ['Stroke Mimic', '31-65', 'Severe', 0.3],

    ['Ischemic Stroke', '65+', 'Severe', 0.3],
    ['Hemmorraghic Stroke', '65+', 'Severe', 0.6],
    ['Stroke Mimic', '65+', 'Severe', 0.8]],
    [StrokeType, PatientAge])

s1 = State(PatientAge, name = 'PatientAge')
s2 = State(CTScanResult, name ='CTScanResult')
s3 = State(MRIScanResult, name ='MRIScanResult')
s4 = State(StrokeType, name = 'StrokeType')
s5 = State(Anticoagulants, name ='Anticoagulants')
```

```
s6 = State(Mortality, name='Mortality')
s7 = State(Disability, name='Disability')

model = BayesianNetwork("Diagnosing")
model.add_states(s1, s2, s3, s4, s5, s6, s7)
model.add_transition(s2, s4)
model.add_transition(s3, s4)
model.add_transition(s4, s6)
model.add_transition(s5, s6)
model.add_transition(s4, s7)
model.add_transition(s1, s7)

model.bake()
p1 = model.predict_proba(
    {'PatientAge': '31-65', \
    'CTScanResult': 'Ischemic Stroke'})[5].parameters[0]['True']
print('p1 = ')
print(p1)
print()

p2 = model.predict_proba(
    {'PatientAge': '65+', \
    'MRIScanResult': 'Hemmorraghic Stroke'})[6].parameters[0]['Moderate']
print('p2 = ')
print(p2)
print()

p3 = model.predict_proba(
    {'PatientAge': '65+', 'CTScanResult': 'Hemmorraghic Stroke', \
    'MRIScanResult': 'Ischemic Stroke'})[3].parameters[0]['Stroke Mimic']
print('p3 = ')
print(p3)
print()
```

```
p4 = model.predict_proba(
    {'PatientAge': '0-30'})[4].parameters[0]['Not used']
print('p4 = ')
print(p4)
print()
```

5 Results

```
P(Alarm) =
0.002516442000000935

P(J&&~M) =
0.050054875461

P(A|J&&~M) =
0.013573889331307633

P(B|A) =
0.3735512282818995

P(B|J&&~M) =
0.0051298581334013

P(J&&~M|~B) =
0.049847949000000002
```

Figure 1: Burglary

```
p1 =
0.5948499999999999

p2 =
0.2600000000000001

p3 =
0.10000000000000044

p4 =
0.5
```

Figure 2: Diagnosing